WE'VE SEEN THIS BEFORE – BALANCING THE COST OF AI

The mainframe industry has always struggled with balancing resource costs against developer productivity. Today's debate about the cost of generative artificial intelligence (GenAI) mirrors past discussions about developer practices and compile time, but with much higher stakes for business competitiveness.

Compiles were once considered expensive, so a developer would sit at their desk for an hour or more reviewing their code to make sure that all errors were resolved before compiling again. CPU time for compiles was measured, but the time developers spent reviewing was not. Lists were kept of CPU time used by developers, who were prized for not using the CPU, which meant they would instead spend time reviewing the code and trying to work out their issues without any assistance.

Over time, as the cost of CPU time dropped, it was realized that developer time was more expensive. Developers felt okay about running more compiles. They would make a reasonable faith effort to correct things, but not put hours into it—the balance changed. This also applied to debugging. A developer would be freed to use a debugger to quickly resolve an error rather than trying to "play computer" at their desk.

I see the same thing happening with AI. It is a line item in the budget, which many see as being very expensive, but they do not consider how it helps the developer and lowers their cost. Currently, developers spend a lot of time reviewing code and making guesses, and that is accepted; it does not have a line-item cost, but an AI server does.

The value of AI must be demonstrated, and its cost is worth it for increased developer productivity.

Developer time, the cost of issues, and being late to market are all more expensive than server time. Consider these real-world scenarios where AI-powered development tools quickly prove their worth:

- A developer using AI to understand unfamiliar code can save hours of investigation time.
- Teams can document code more effectively with AI-generated explanations.
- Issues can be resolved faster through AI-assisted debugging.
- A developer doing an AI-generated explanation of unfamiliar code can easily save it as a comment in the program to help the next developer.

These examples would incur a one-time cost that would pay off both immediately and over time. Obtaining cost information can be the challenging part of this equation. How do you measure time to balance it against the cost of AI? BMC AMI zAdviser can help by providing metrics on the current state of mean time to resolution (MTTR) and the time spent on new features. This helps to calculate developer cost, which you can balance against the cost of AI to assist.

This is where BMC AMI zAdviser Enterprise proves invaluable. Through its comprehensive analytics dashboard, it provides concrete metrics that help quantify both sides of the AI investment equation:

- Development velocity metrics that show time spent on code review and understanding
- Lead time measurements that reveal delays in feature delivery
- Quality metrics that demonstrate the cost of errors and technical debt

Making the case for AI investment requires both historical perspective and forward-looking metrics. BMC AMI zAdviser provides the data-driven foundation needed to:

- Establish current baseline metrics for development speed and quality
- Track improvements as AI tools are implemented
- Demonstrate concrete return on investment (ROI) through reduced development cycles and faster feature delivery
- Guide ongoing optimization of AI resource allocation

Just as mainframe teams eventually embraced more frequent compiles for better productivity, today's organizations are discovering that the business value of AI-powered development far outweighs the infrastructure costs—and BMC AMI zAdviser helps prove it with actual data.

Learn more about BMC AMI zAdviser and its new on-premises version in the on-demand BMC AMI Tech Talk, Introducing On-Prem BMC AMI zAdviser.